PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/46, 1/00 — A1

(11) International Publication Number: WO 98/08163

(43) International Publication Date: 26 February 1998 (26.02.98)

(21) International Application Number: PCT/IB97/00973

(22) International Filing Date: 7 August 1997 (07.08.97)

(30) Priority Data:

9616783.8	9 August 1996 (09.08.96)	GB
9703773.3	24 February 1997 (24.02.97)	GB

(71) Applicant (*for all designated States except US*): APM LIMITED [GB/GB]; Poseidon House, Castle Park, Cambridge CB3 0RD (GB).

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): BULL, John, Albert [GB/GB]; The Almshouses, Great Brington, Northants NN7 4HY (GB). OTWAY, David, John [GB/GB]; 12 Willis Road, Cambridge CB1 2AQ (GB). KRAMER, Andre [GB/GB]; 16 Clare Street, Cambridge CB3 4BJ (GB).

(74) Agents: DUMMETT, Thomas, Ian, Peter et al.; Dummett Copp, 25 The Square, Martlesham Heath, Ipswich, Suffolk IP5 3SL (GB).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: ISOLATED EXECUTION LOCATION

(57) Abstract

The present invention provides an end user computer system programmed to operate in response to an imported data stream containing or having associated therewith one or more mobile program components from an external source, characterised in that: a) the incoming data stream is screened to identify mobile program components within or associated with that data stream; b) a selected some or all of the mobile program components are passed to one or more program execution locations selectively isolated from or within the end user system prior to being executed to operate in a desired manner; c) the execution location is one in which one or more of the selected program components are retained and which has one or more interfaces with the external source of the data stream and one or more interfaces with the end user system whereby program component(s) within the execution location can be executed within the execution location to interact with the external source of data and/or the data and/or a program held by the end user system; and d) the operation of the interface(s) between the execution location and the end user system are programmed so that only data which has been interacted on by the program component(s) within the execution location in a specified and controlled manner and/or program components which operate in a specified manner can be passed to and from the end user system.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

TITLE: **ISOLATED EXECUTION LOCATION**

The present invention relates to a method and apparatus, notably to a method for selectively directing portions of an incoming stream of data from an external source to a selected location at which program components within that incoming stream can be interpreted and executed. This reduces the risk of corruption or damage to data or programs held in an end user network of computer systems by mobile program components imported into that system from an external source. The invention also enables large program components in the incoming data stream to be interpreted and executed on a specific server and thus reduce the load on the processors in the downstream end user computer system. This will reduce the need for large processing capacity individual terminals in that down-stream system. The invention also relates to a computer system programmed or modified to operate using the method of the invention.

BACKGROUND TO THE INVENTION:

The user of a stand alone or networked computer system, both hereinafter collectively called an end user system, often has a need to import data or whole programs or parts of programs, hereinafter collectively and individually called program components, from an external source to enable the end user system to operate in a desired manner. The term program component is used herein to denote material which is interpreted by a computer system to operate the system in a specific manner, whereas data is the information material upon which the computer system acts under the direction of the program components.

Users, particularly banks and other financial institutions, need to be able to inspect incoming program components and data to ensure that those components will

CONFIRMATION COPY

not corrupt or otherwise adversely affect the data and operating programs held in their end user systems. This is particularly important where there are a number of computer systems operating within a network and in which the network or computer systems within that network have a number of points at which access to external data and program sources can be made.

Where the system is a closed network and the data or program components are provided from other units within the same network, the user can satisfy himself that the data and program components do not contain material which could adversely interact with the data and programs held on his particular unit of the network. Such satisfaction will typically involve the inspection of the computer listings upon which the program is based to ensure that there are no errors or adverse components within the program. However, where the program or program component being imported is large and complex, such inspection and verification becomes excessively time consuming and expensive and therefore impractical.

Where the end user network or stand alone system is to receive data or program components from a source which is external to the network, for example from the Internet or an external data base, the risk of deliberate or accidental introduction of program components which can adversely interact with the data or programs already held in that end user system is increased. Since the external data source may be operating under one or more large and complex programs, which are themselves under continual updating and revision, it is effectively impossible to inspect each program and every modification of the program to ascertain that the end user system will not import adversely acting material.

There are a number of forms of program components which

can cause corruption or damage to data or programs held in an end user system and/or can cause other deleterious effects when imported into the end user system. Such program components include those which are deliberately designed to corrupt the data or operating programs of the end user system; those which collect confidential data from the end user system and transmit that data to an external location without the end user being aware that such unauthorised transmission or theft of data is taking place; and programs which deny the user full and proper use of the end user system, for example by introducing repeating closed loop operations which consume the computing capacity of the end user system or deny access to areas of the end user system. Such program components are known as viruses, zappers, hostile Applets, Trojan Horses and service deniers and will be generically denoted as viruses hereinafter. A widespread concern is the deliberate distribution of such virus programs or program components into an end user system where they are executed and adversely interact with or cause disruption to the proper operation of the system. Such viruses may not be intentionally damaging, but may be capable of causing damage and/or economic loss inadvertently. Whilst an end user can repeatedly inspect data and programs within a closed network to detect such viruses, the end user cannot inspect the external data or program source for such viruses and must accept the risk that any import of data or program components from an external source may import viruses into his system.

In order to reduce the risk of importing material from an external source which could adversely interact with an end user system, it is commonplace to screen all incoming data and programs or program components to identify the source of that material. Only material from specified sources is permitted access into the end user system. The end user can satisfy himself that such sources provide data

and/or programs which have been inspected either by the end user or by the source. Alternatively, the user can base his confidence in the source of material on its reputation for accuracy in compiling programs and for reducing the presence of possible adversely acting program components in any program components it makes available to end users. Such screens are known in the computer field as firewalls and act uni-directionally on a communications hardware level to allow incoming material to pass if it comes from a specified communication address or to destroy incoming material if it does not come from a specified source.

However, a firewall prevents access by the operator of an end user system to data and programs which are from non-specified sources. This restricts the freedom of the user to access alternative sources until they have been inspected and authorised. Furthermore, where the external source is operating under a large or complex program, such inspection is not practical and any authorization of access to that external source may destroy the integrity of the end user system.

These problems are aggravated where there are many points in the end user system from which external sources of data and programs can be accessed. It has been proposed to limit the number of such access points in an end user system and to ensure that all incoming material is fully screened at the permitted access points. This can be done by diverting the incoming material to a location, for example one known as a web proxy, at which the system manager can inspect it before it is passed to the end user system. However, this can lead to excessive bottle necks in the operation of the system and delays in accessing the external source from any given end user computer system in a network. Furthermore, inspection of the incoming material may not identify program elements which, whilst

intended to operate benignly, could operate errantly. These are often passed direct to the end user system, by-passing the web proxy, where they are implemented with potentially deleterious effect. Additionally, where acceptance of incoming data is based upon identification of the source of such data as an accepted source, this places undue trust in the integrity of that source and its ability to generate material without errors or problems.

It is also common place to provide one or more virus detection programs within an end user system. These operate by recognising characteristic patterns in the virus program and destroying the virus program before it is executed within the end user system. The detection program may also recognise specific sites in an operating program to which a virus may attach and remain dormant until executed and thus detect when a virus is present by a change in such a site. However, this requires that the detection program recognise specific features or patterns and requires that the virus be imported into the end user system before it can be identified and neutralised. Furthermore, where the virus is one which is not recognised by the virus detection program, for example because it is a new virus or a mutation of an existing one, the virus may not be detected and may be executed within the end user system.

In many applications it is desirable to provide program components from the data source to enhance the execution of programs held within the end user system. For example, many data sources written in the JAVA or JAVASCRIPT languages utilise mobile program components, or Applets. The program components can be included within the stream of data from an external source to enable the end user system to handle the data effectively. For example, the Applet can enable the end user to interact with the external data source in real time to perform a scripted

sequence of actions, for example to achieve animation of graphic images upon the end user system video screen from data already held at the end user system. This avoids the need to transmit the data for each image from the external source and thus speeds up the operation of the end user system. Alternatively, such program components are retained at a specific address in the external source and the incoming information stream contains a characteristic flag or other identifier which alerts the end user system that it needs to download a program component from the external source and the address from which that program component should be downloaded. Such program components are hereinafter referred to as being associated with the incoming data stream from the external source. The operator of the end user system downloads the required program component into the end user system where it is implemented.

Such program components are termed mobile since they are intended to be imported into the end user system and to be executed within that system and to interact in a beneficial manner with the data and program components held at the end user system. It is therefore necessary that they should be accepted by the end user system. They therefore pass through any firewall and are not rejected or destroyed by a virus detection program. It has been proposed to sign and seal such program components cryptographically so as to identify the program component as coming from an authorised source, for example one where the components have been individually inspected. However, this requires the end user to place complete trust in the integrity and competence of the organisation cryptographically signing and sealing the program components they export.

Where the program or program component imported from the external source is large and complex and/or is constantly

being updated, as is the case with network browser programs, it is not possible to provide a high level of confidence in such programs or program components. This may present an acceptable risk to the operator of the end user system when balanced against the advantages that the use of such program components gives.

Furthermore, it is possible that such mobile program components, whilst satisfying the authentification or identified source criteria, can be interpreted incorrectly in the end user system and/or can deliberately or accidentally interact adversely with the data and/or program components already held by the end user system. This raises a problem for the end user operator. On the one hand, the importation of the mobile program components is desirable for the proper operation of the end user system; but they can cause corruption of data and damage to the operating and other programs held by the end user system. The conventional firewall or virus detection programs cannot protect the end user system without preventing proper operation of the system. As stated above, diverting the imported data stream to a holding location, for example the terminal operated by an end user system manager, where any program components in the data stream are inspected to establish that they are acceptable to the end user system before they are passed to the end user system to be implemented, cannot discriminate between wholly benign program components and those which could operate errantly.

The problem of deliberate or accidental errant interaction of desirable mobile program components from an external source has been recognised as a major problem by the computer industry, but no effective solution has yet been proposed.

We have now devised a method and apparatus by which an end

user system can be protected from the errant effects of such otherwise desirable mobile program components imported from an external source. The method of the invention can also be used to selectively direct program components from an incoming data stream to a server specifically designated to run that program component or type of component. In this way, large program components can be executed in that server isolated from the remainder of the end user system and the results of that execution transmitted to the end user system. This enables large and complex program components to be executed in a server dedicated to this purpose and avoids the need for the end user to provide large and complex terminals in his end user system capable of executing these program components. Moreover, a plurality of selected program components from the same incoming data stream can be directed to different execution locations isolated from one another so that the speed and security of handling different types of program components is enhanced. By selecting the server at which a specific type of program component is executed, it is possible to tailor make the operation of that server to the program component it is to execute and thus be more specific in the security measures or policies which that execution location provides to the end user system.

SUMMARY OF THE INVENTION:

Accordingly, the present invention provides an end user computer system programmed to operate in response to an imported data stream having one or more mobile program components from an external source contained in or associated with the data stream, characterised in that:

a. the incoming data stream is screened to identify mobile program components present in or associated with that data stream;

b. a selected some or all of the mobile program components are passed to one or more program

execution locations selectively isolated from, or isolated within, the end user system prior to being executed to operate in a desired manner;

c. the execution location is one in which one or more of the selected program components are retained and which has one or more interfaces with the external source of the data stream and one or more interfaces with the end user system whereby program component(s) within the execution location can be executed within the execution location to interact with the external source of data and/or the data and/or a program held by the end user system; and

d. the operation of the interface(s) between the execution location and the end user system are programmed so that only data which has been interacted on by the program component(s) within the execution location in a specified and controlled manner and/or program components which operate in a specified manner can be passed to and from the end user system .

In a preferred embodiment of the invention, a stand in replacement for the program component in the incoming data stream, known for convenience as a proxy Applet, resides in the end user system and receives and acts on the data from the execution location. The proxy Applet mimics the actions of the isolated selected program component on the end user system and can be in the form of an accessory to the browser program and can be provided in the same language as the program component that it mimics so that it can be viewed by the browser program as if it were the program component it mimics without the need for extensive modification of the browser.

The isolated execution location presents the same interfaces to the isolated program component as the end user system presents to the proxy Applet so that the

operating environments for the two are substantially identical. Operating requests made to the interfaces within the execution location by the isolated program component are transmitted over a communications protocol to the proxy Applet residing in the end user system which re-issues those operating requests to the identical interfaces on the end user system. In this way only a sub-set of the end user interfaces which have been fully inspected and verified are made available to the execution location and in a controlled manner. The data streams mediating those operating requests can also be screened to detect deviations from a strict, well specified and verified specification.

In the present invention the execution location can be located upon the same physical site as part or all of the end user system, for example associated with an access gateway to the end user system. Alternatively, it can be located remotely from the downstream remainder of the end user system. Moreover, the execution location need not be located within the end user system itself, but can be located as a protective isolation screen between the external source of the data and program components and the downstream end user system. For convenience, the term "within the end user system" will be used herein to denote the case where the execution location is provided within one or more of the computer units of the end user system; and the term "external to the end user system" will be used to denote a computer unit or other means which, whilst it may be located on the same geographic site as the end user system, is isolated from the end user system.

By providing the execution location isolated from the remainder of the end user system, incoming mobile program components are contained selectively isolated from or within the end user system. Although the program components may operate errantly within the execution

location, they are only permitted to interact with the end user system in a specified and controlled manner, for example via a general utilities interface operating to pass only selected data. Since the program operating the execution location can be comparatively small, the end user can inspect the program listing for that program to ensure that it fulfils the desired criteria. The operator can thus have a high level of confidence that only data which has been processed in a specified desired manner or a program component which operates in a desired manner can be transmitted from the execution location to the end user system. The operating program for the execution location can also prevent the passage of program components to the end user system from the execution location, thus minimising the risk of viruses or undesirable program components entering the end user system. In addition, the execution location can operate to limit access of a program component from the execution location to specified resources within the end user system, for example to limit access time to the central processor to minimise the effect of a services denial virus, to limit the bandwidth of the communications access, or to limit access to certain disc blocks in the hard disc memory.

The program operating the execution location can be selectively written so as to permit transmission or reception of data only to or from specific sources within the end user system and/or a specific external source, so that imported program components executed in the execution location cannot access certain areas of the data base in the end user system. For example, an execution location could be programmed only to operate on purchase ledger data, another could be programmed to operate only on personnel data. It may therefore be necessary to provide a series of execution locations, each designed to operate in an individual manner on specified data sources and destinations. This will enable the operating program for

each execution location to be smaller and more specific and hence easier to inspect and verify. Furthermore, it is within the scope of the present invention to provide a number of tiers of execution locations operating in series with one another so that the incoming data stream is directed to an initial execution location at which the identities of the program components, their sources and/or their functions can be identified. This initial execution location can then select the execution location(s) in the next tier to which specified program components are directed and so on. In this way progressively more stringent security requirements or selection of the appropriate portion of the end user system to utilise the program components from the incoming data stream can be applied as a series of simple steps and/or large program components can be isolated and directed to execution locations specifically configured to execute those program components.

For convenience, the invention will be described hereinafter in terms of an execution location programmed to operate with a single external source or destination of data external to the end user system and to transmit or receive data to or from a single destination within the end user system. However, it will be appreciated that the invention can be applied to execution locations operating with a plurality of external and/or end user sources and destinations in any combination.

The execution location is selectively isolated from or within the end user system, that is the execution location can only receive and transmit data and/or program components in a selective and controlled manner via the interfaces with the external source and the end user system. Thus, the execution location will typically require a level of intelligence and data storage so that it can accept and store the incoming mobile program

components from the external source and can then execute those program components under the control of an operating program already held within the execution location to interact with data from the external source and/or from the end user system. If desired, the operating program required for the execution location can be held within the end user system to minimise corruption from external sources, and is transmitted to the execution location as part of the start up procedure of the end user system. As explained below, several operating programs may be available to the execution location depending upon the type of data it is to handle and the type of operation to be carried out on that data. The end user or system administrator may be provided with means, for example specific keyboard operations, which load the appropriate operating program to the execution location where a selection has to be made between various alternatives.

The requisite processing and data storage functions for the execution location can be provided by one of the computer units within the end user system network or as an isolated portion of one of the computer units, so that the execution location is located within the end user system. However, with current computer architecture it may not be possible to provide a sufficiently isolated environment in which the program components are executed. It is therefore preferred to provide the execution location as a separate physical unit selectively isolated from the end user system with which it is to interact and to provide limited access routes or interfaces between the execution unit and the end user system which are operated under the control of the program operating the execution location unit. This allows conventional firewall and communications protocols to be used to separate the execution location from the end user system. Typically, the execution unit will be a conventional computer having its own processor and memory capability.

For convenience, the invention will be described hereinafter in terms of a separate computer acting as an execution location which is physically separate from the end user system, but which is configured as if it were a data import access point to the end user system network.

Whilst the execution location will preferably transmit only data to and from the rest of the end user system, it may be programmed to permit transmission of program components to the end user system. Since such program units will have been subjected to inspection within the execution location by a program which the end user has inspected or in which he has a high level of confidence, the risk that such transmitted program components will operate errantly within the end user system is minimised. The execution location can thus be used to inspect and verify incoming program components which it is desired to download into the end user system and can be used to intercept virus programs before they reach the end user system. Whilst the end user may be confident that specified program components are acceptable, in which case he may feel that they do not need to be inspected within the execution location and can be passed directly to the end user system, this exposes the end user system to possible errant operation of those program components. It is therefore preferred to elect to pass all program components within or associated with an incoming data stream to the execution location.

For convenience, the invention will be described hereinafter in terms of the transmission solely of data to the end user system.

As stated above, the execution location transfers data between the execution location and the end user system only if it complies to predetermined criteria so that the execution location regulates the exchange of data to and

from the end user system and the execution location. The operating program required to achieve this and to identify the existence of program components in or associated with the data stream from the external source can be written using conventional programming techniques having regard to the source and destination within the end user system required for the initial data and the resultant processed data. Whilst the program operating the execution location may be retained and implemented wholly within the execution location, it is within the scope of the present invention to locate part of that operating program at some other location. Thus, the portion of the program which identifies a program component in the incoming material and diverts that to the execution location (the snare) may be located at each operating unit of the end user system, for example as part of the browser program, or can be incorporated in the operating program for the firewall(s). However, location of the snare program in the browser program requires that each operating unit in the end user system be provided with the necessary snare program; and that any extension of the end user system or variation of the browser program may require re-programming of the whole end user system to ensure that the snare program is present at all end user operating units. It is therefore preferred to incorporate the snare component for each data stream to be received from an external data source in the operating program to be operated within the execution location.

As stated above, a single execution location can be used to achieve a specified operation upon specified data. However, it is within the scope of the present invention to provide an execution location which can operate upon several categories of information and/or with data from several sources and/or destinations of information. This may require separate operating programs for the execution location to run concurrently or consecutively. Alter-

natively, separate execution locations can be provided, each to achieve a specific operation upon specific data. It will also be appreciated that one or more execution locations can be provided at each point at which the end user system accesses an external data source and that each such access point can have an execution location which is to operate in a specific manner upon specific data using selected program components from the external data source. The operator of the end user system can thus identify the function and potential sources to be accessed at each access point by selection of the operating program under which the execution location at that access point operates. This will further regulate the importation of material from external sources to the end user system.

It is also within the scope of the present invention to create zones within a system into which access from other zones of the same system or network is regulated by providing execution locations of the invention at the access points between the zones of the system, one of which is deemed to be the external source of information and the other the end user system of the invention.

Whilst the execution location of the invention may provide the sole regulation of importation of material into an end user system from an external source, it is preferred to locate the execution location between two firewalls which inhibit transmission of undesirable program components and/or data from the external source and/or the end user system. Such firewalls can be of conventional form and serve to reduce the load imposed upon the execution location by providing primary control of the flow of material to and from the end user system. As indicated above, the snare component of the operating program for the execution location may be incorporated in the operating program for either of both of the firewalls. However, to protect the snare program from external

corruption or attack, it is preferred that the snare program is not located on the external source side of the firewall located between the external source and the execution location.

The invention is of especial application to data sources operating under JAVA or JAVASCRIPT technology which utilise mobile program components or Applets. These are executed by the end user system to customise the application program under which data is to be processed to suit the end user system and to set up the end user system for the receipt of data from the external source. However, the invention is applicable to the interception and storage in the execution location of program components from other computer languages and operating systems, for example ActiveX, perl, tcl/tk, c, c++ and sh and its variants.

Where a program component within an execution location has operated in an undesirable manner, the results of that errant operation are retained within the execution location and are not allowed to be transmitted to the end user system or the external source. The detection of an errant operation within the execution location can cause the execution location to close down and re-initialise so as to delete and re-instate all data and program components within the execution location. If desired, the program component causing the errant operation can be identified for audit purposes to identify the source of the program component. The end user can thus investigate the integrity of the source and take appropriate action regarding importation of further data from that source.

The deletion of material from the execution location and re-initialisation or other subsequent operations can be carried out using conventional programming techniques. The execution location(s) can thus be viewed as

sacrificial and can be re-initialised without affecting the remainder of the end user system, thus avoiding major re-programming and data re-instatement which would otherwise be required if the program component had been held within the end user system.

The invention has been described above in terms of an end user system receiving data from an external system which is accessed by a number of other users, that is a public access system or service provider. However, the invention can also be applied to the public access service provider so as to protect the public access data base from corruption by importation of undesirable program components, for example viruses. In this case, the execution location is provided at some or all of the access points to the service provider in a similar manner to that described above for the end user system.

Furthermore, it is possible to carry out the detection and separation of mobile program components from a data stream with which the mobile program components are to interact at source. Those program components can then be transmitted separately to the isolated execution location of an end user, where they can be implemented to execute functions on the remainder of the data stream with which they are associated. Such program components can be normally resident within the execution location and need not be downloaded from the external data source each time that external data source is accessed.

Accordingly, in another form the invention provides an end user computer system programmed to operate in response to an imported data stream from an external source and in response selectively to one or more mobile program components associated therewith, characterised in that:

a. the selected mobile program component(s) are fed to and/or retained in one or more execution locations,

which are selectively isolated from or within the end user system, prior to being executed to operate in a desired manner; and

b. the execution location is one in which one or more of the program components are retained and which has one or more interfaces with the external source of the data stream and one or more interfaces with the end user system whereby program component(s) within the execution location can be executed within the execution location to interact with the external source of data and/or the data and/or a program held by the end user system; and

c. the operation of the interface(s) between the execution location and the end user system are programmed so that only data which has been interacted on by the program component(s) within the execution location in a specified and controlled manner and/or program components which operate in a specified manner can be passed to and from the end user system.

Preferably, the incoming data stream is operatively associated with the mobile program components, for example the data stream contains the program components or contains means for identifying the program components held in the external source to be implemented in association with that data stream, and is screened to identify the mobile program components within or associated with that data stream; and selectively some or all of those mobile program components are passed to one or more program execution locations selectively isolated from or within the end user system prior to being executed to operate in a desired manner.

The invention also provides a method for operating an end user computer system, which method comprises importing from an external source into the end user system a data

stream containing or having associated with it one or more mobile program components which it is desired to execute on the computer system, which method comprises:

a. executing the selected mobile program component(s) within one or more program execution locations selectively isolated from or within the end user system so as to interact with data from the end user system and/or from the external source;

b. passing program components which operate in a specified manner and/or the resultant data from such interaction to the end user system via an interface which permits the transmission to or from specified locations in the end user system and/or in the external source of data and/or program components which correspond to specified criteria.

Preferably, the program components are operatively associated with the data stream transmitted to the end user system and the method includes the steps of screening the incoming data stream to identify mobile program components within or associated with that data stream; and passing some or all of the mobile program components to the isolated execution location. Preferably, the program components which are passed to the execution location are selected as those required for the execution of the desired part of the incoming data stream, thus distinguishing the invention from other systems in which the whole of the incoming data stream is down loaded and a selection of the relevant portions made within the end user system.

The invention has been described above in terms of incoming data from a web site. However, it can be applied to incoming data from any other form of external data source, for example e-mail or other message based information transfer systems. Thus, the invention can be applied to communications between two private network

systems or between elements of a single network system, either directly or via an external domain or other server.

The invention has been described above in terms of security provided by a single layer of execution locations. However, it is within the scope of the present invention to employ two or more layers of the execution locations whereby only data and/or program components which have been screened by a previous execution location are passed to a succeeding execution location for further screening before the program component is allowed to interact with the end user system. Furthermore, the nature of the operating programs in successive layers can be radically different so that different functions can be achieved in successive layers and program components can be subjected to differing types of security investigation. Furthermore, the ability to implement more than one mobile program component in a single execution location gives the operator flexibility in the handling of incoming program components either alone when they reside in separate execution locations, or in inter-action with one another where they co-reside in an execution location. Since the isolated execution locations are operated in isolation from the end user system, in the event that a program component within an execution location operates errantly or is detected as potentially damaging, the execution location can be viewed as sacrificial and the operator can re-initiate an execution location in isolation from other execution locations or the end user system without the need to re-program the whole end user system.

As stated above, the invention can also be used to control the flow of the incoming data stream and to selectively direct part or all of the stream and its associated program components to specific execution locations. In this way large program components can be identified and selectively directed to an execution location specifically

designated to execute such large program components. The results of the execution of that program component can then be transmitted to the end user system where the terminals of that system need not each have the computing power required to execute the large program component. Alternatively, the invention can be used to selectively direct parts of the data stream and their associated or contained program components to specific execution locations designated to execute that type of program component. This ability to identify and route selected portions of the incoming data stream before the program components are executed in the execution location further reduces the need for the end user system administrator to inspect every incoming program components at the entry point to the end user system and allows that inspection to be carried out on a reduced volume of incoming material at a lower level in the end user system.

DESCRIPTION OF THE DRAWINGS:

The invention will be described by way of illustration only with respect to the preferred embodiment of the invention as shown in the accompanying drawings, in which Figures 1 and 2 show in diagrammatic block form a typical present method of operating an end user system to receive a data stream from an external source; Figures 3 and 4 show a system operated according to the method of the invention; and Figures 5 to 9 show in block diagram form a system operating using JAVA language.

DESCRIPTION OF THE PRESENT SYSTEM:

Figure 1 shows a current system for importing data and/or program components from an external source, for example data from a Web server, and comprises a single computer unit or a plurality of computer units in an end user computer system interconnected by a private network, and

the Web server connected to the public network. The end user uses a browser or other program held on the end user system to identify the data and/or program components which are to be imported from the Web server. The browser program may be held on any or all of the computer units in the end user network. In order to reduce the risk of importing undesirable material, the access point to the end user system is provided with a firewall which only allows the passage of data and program components from the public network which come from or go to specified addresses in the public network. The browser and/or firewall may also refuse to import program components or data that have not been cryptographically signed and sealed by a known and/or trusted source. It is also customary to hold a virus detection program on the end user system which detects the characteristic patterns of known viruses or changes which such viruses make in programs carried by the end user system.

However, where the data stream from the public network contains mobile program components which it is desirable to execute on the end user system, problems arise in ensuring that the imported material does not contain material which could operate incorrectly or maliciously on the end user system. The firewall will allow such program components to pass since they come from an accepted address. Whilst a firewall may have a measure of intelligence, it operates by identification of the communication address from or to which the data and program components are transmitted. The virus detection program cannot guarantee to detect and de-activate the program components when they operate incorrectly since they may not be recognised as detrimental viruses. As a result, the end user system is vulnerable to importation of errant program components.

Such a problems exists specifically with systems operating

using JAVA technology as exemplified by the system shown in Figure 2. The end user, identified as the host platform, carries a browser program which is used to request data from a service provider. This is provided via the network server which transmits a stream of data containing JAVA Applets, which the end user desires to run on the end user system. The JAVA Applets are executed on the end user system via a JAVA Virtual Machine which maintains the interfaces to the end user system via the browser program and to the Web server. However, the browser may incorrectly interpret the Applets or the Applets may accidentally or deliberately contain harmful instructions. At present there is inadequate protection for the end user system from such errant interpretation or harmful instructions.

DESCRIPTION OF THE PREFERRED EMBODIMENT:

In the method of the invention as shown in Figure 3, data from the public network is not fed directly to the end user but is scanned and any program components within the data stream are diverted into an execution location, denoted as a cage. The scanning of the incoming data stream to identify program components in it and to divert those to and execute those in the cage can be done by any trusted component located in the path of the data stream, for example a local Web proxy, the browser program, a network router or a dedicated program. The execution cage is typically a conventional commercial computer interfaced between the end user system and the public network.

The execution cage thus acts as a protective screen between the public network and the private network of the end user and by virtue of the program under which it operates selectively and controlledly permits the transmission of data and/or program components which meet specified criteria, for example come from or are addressed

to specified locations in the end user system or the external source.

In order to enhance the confidence of a user in the operation of the cage, it is preferred to provide firewalls at the interface between the cage and the end user system and between the cage and the public network. Either or both of these firewalls may be incorporated into the cage if desired. That portion of the overall system shown in Figure 3 downstream of and including the outer firewall is located at the end user site. That portion downstream of and including the inner firewall is the end user system which is protected by the execution cage.

With reference to the system shown diagrammatically in Figure 4, the Applets and other program components are executed within the cage. Data from the end user system and/or from the public network is interacted upon by the program components in the cage and the resultant modified data is transmitted to the end user system via a suitable interface. The interface is programmed to transmit data which satisfies specific access criteria but does not transmit or receive program components or data which does not satisfy the access requirements. Therefore, the cage retains the program components selectively isolated from or within the end user system and strictly regulates the data which is transmitted to and received by it. As a result, the end user system imports and exports only data which is acceptable. Since the program(s), in this case those operating the JAVA Virtual Machine, which control the operation of the cage can be small, they can be readily inspected. The end user can thus verify the integrity of the program(s) to his satisfaction and can ascertain that the program is clearly and logically constructed and well documented by inspection of the program listing. Those components of the interface between the JAVA Virtual Machine and the public network

which regulate the import of the JAVA Applets into the cage are also inspected to ensure the integrity of the operation of the cage.

Once within the cage, the Applet is executed under the control of the JAVA Virtual Machine so as to interact with data from the public network and/or the end user system in a controlled manner. In order to contain errant operation of the Applet and prevent it from gaining access to the end user system, some form of access control is provided either in the end user system or in the cage. Such access control can be in the form of an internal firewall. Access control can prevent the Applet from accessing any source or destination other than ones on a list provided by a system administrator. Alternatively, the access control can require positive consent from the browser user before each access request is allowed to proceed. The program components within the cage can be classified into those components which must be inspected and verified in order to ensure the integrity of the cage, and those which can safely be imported and executed under the control of the former. The dividing line between these two sets of program components is known as a security membrane.

The necessary hardware for the cage can be of conventional nature and the operating program(s) can be prepared using conventional programming techniques and algorithms.

A particularly preferred method of operating the invention with a source of information in the JAVA language is shown in Figures 5 to 9. As shown in Figures 5 and 6, a JAVA powered Web server is accessed by an end user via a firewall, using a browser program. Information from that Web server is downloaded to a Web proxy where the data is held before it is transmitted to the end user system. The incoming information stream contains one or more flags or

other identifying features which are recognised by the Web proxy as either identifying the presence of a JAVA Applet in the incoming data stream or identifying the site address in the Web server of a JAVA Applet required to be run on the end user system to display the data from the Web server. In a conventional system, the end user would download the JAVA Applet from the designated site on the Web server directly to the JAVA Virtual Machine in the browser program in the end user PC terminal or other operating unit in the end user system. The JAVA Applet would then be implemented within the end user system to operate on the data held in the Web proxy or in the end user system data store to provide the desired image on the display screen. However, if the Applet does not operate correctly, the end user system has not protection against this since the Applet is operating within the end user system.

In the system shown in Figures 7 to 9, the execution location, or Cage server, is located on the public network, shown as the Internet, side of the inner firewall and there is typically another, outer firewall, not shown, located between the Cage server and the public network. The Cage server is typically a computer having processing and memory capacity which receives and interprets program components which it receives from the public network. Located between the inner firewall and the end user operating unit is a Web proxy, which is typically another computer or part of the same computer as the Cage server. The Web proxy receives and stores the data from the public network, designated as the HTML page. The Web proxy preferably contains the program component which identifies the presence of a flag or other indication in the incoming information which identifies the presence and address of an Applet required to interact with the data in the incoming data stream from the public network and/or from the end user data store to give the required display on

the end user operating unit.

This snare program component identifies the address of the Applet in the Web server and diverts any incoming Applet to the Cage server where the incoming or real Applet is implemented in isolation from the end user system. The Snare program also generates a new or translated address to identify that real Applet to the end user system, which new address corresponds to the address of a proxy Applet held within the browser program in the end user operating unit or PC. The proxy Applet is one which the end user has written or obtained and which has been fully verified so that it can operate safely within the end user system.

When the end user is notified by the browser program that an Applet requires to be downloaded from the Web server, the snare program will be give the new address as the address for that Applet. As a result, the proxy Applet and not the real Applet will be addressed and implemented within the end user system. However, the proxy Applet will interface with the Cage server to download the real Applet from the Web server into the Cage server. The real Applet can now be run within the Cage server in isolation from the end user system and interfaces in a controlled manner via a general utilities interface (GUI) to act upon the environment of the proxy Applet within the end user browser. The real Applet does not download into the end user system as with a conventional browser operation.

Thus, the invention also provides an end user system in which the external data source operates under JAVA or JAVASCRIPT language and the program components which are to be imported and run within the execution location are Applets; and in which the program for operating the execution location and/or the end user system identifies the address of the Applet to be imported and run in the execution location and translates that into a new address

corresponding to a proxy Applet held within the end user system and adapted to be implemented within the end user system and to interface within the imported Applet in the execution location to implement the latter Applet within the execution location.

From another aspect, the present invention provides a method of the invention in which the incoming data stream is written in JAVA or JAVASCRIPT language and contains or is associated with one or more Applets to be imported to the execution location, and the operating program of the execution location and/or the end user system operating program identifies the address of the Applet to be imported and creates a new or translated address for that Applet corresponding to the address of a proxy Applet within the end user system, whereby when the end user system is actuated to download the imported Applet it will address and implement the proxy Applet to operate on data in the incoming data stream and/or held in the end user system and will interface with the imported Applet in the execution location.

Figures 5 to 9 show the following items:
Figure 5: Java applets embedded in Web Pages; Figure 6: Java applets penetrate the Firewall by downloading executable code into the client; Figure 7: The Cage-Snare transforms the HTML & substitutes a proxy applet into the browser; Figure 8: The proxy applet establishes a connection to a CAGE server process; Figure 9: The applet is loaded into the CAGE and the GUI pipe to the proxy applet established.

CLAIMS:

1. An end user computer system programmed to operate in response to an imported data stream from an external source and in response to one or more mobile program components contained in that data stream or associated therewith, characterised in that:

a. the mobile program component(s) are fed to and/or retained in one or more execution locations, which are selectively isolated from or within the end user system, prior to being executed to operate in a desired manner; and

b. the execution location is one in which one or more of the program components are retained and which has one or more interfaces with the external source of the data stream and one or more interfaces with the end user system whereby program component(s) within the execution location can be executed within the execution location to interact with the external source of data and/or the data and/or a program held by the end user system; and

c. the operation of the interface(s) between the execution location and the end user system are programmed so that only data which has been interacted on by the program component(s) within the execution location in a specified and controlled manner and/or program components which operate in a specified manner can be passed to and from the end user system.

2. An end user computer system as claimed in claim 1, characterised in that it is programmed to screen the incoming data stream to identify mobile program components present in or associated with that data stream and so that a selected some or all of the mobile program components are passed to one or more program execution locations selectively isolated from, or isolated within, the end

user system prior to being executed to operate in a desired manner.

3. An end user computer system as claimed in either of claims 1 or 2, characterised in that the execution location is provided by a computer located intermediate the external source of the data stream and the downstream end user system, which computer is isolated from the downstream end user system and communicates with that end user system via one or more interfaces whose operation is controlled to permit the passage of data and/or program components in a selective and controlled manner.

4. An end user system as claimed in any one of claims 1 to 3, characterised in that the execution location is programmed to operate with a single source and/or destination of data external to the end user system and to transmit or receive data to or from a single destination within the end user system.

5. An end user system as claimed in any one of claims 1 to 3, characterised in that a first execution location receives a plurality of said selected mobile program components and identifies the function and/or character of those program components and directs selected ones of said program components to other isolated execution locations according to the function or character thereof.

6. An end user system as claimed in claim 5, characterised in that said other execution locations are programmed to operate under a different regime to said first execution location so as to provide different executions of the program components.

7. An end user system as claimed in claim 1, characterised in that it is provided with a plurality of execution locations, each programmed to operate with a

different external data source and with a different destination within the end user system.

8. An end user system as claimed in any one of the preceding claims, characterised in that the execution location is provided with one or more firewalls between it and the external source of data and/or the end user system downstream of the execution location.

9. An end user system as claimed in any one of the preceding claims, characterised in that the execution location is provided with program means adapted to operate program components received from the external source in a specific manner and to permit transmission of solely data and/or program components to the end user system downstream of the execution location which satisfy specific predetermined criteria.

10. An end user system as claimed in any one of the preceding claims, characterised in that the external data source operates under JAVA or JAVASCRIPT technology and the program components which are to be intercepted and run within the execution location are Applets.

11. An end user system as claimed in claim 10, characterised in that the program for operating the execution location identifies the address which the Applet to be run in the execution location has and translates that into a new address corresponding to a verified Applet held within the end user system and adapted to be implemented within the end user system and to interface with the Applet in the execution location to implement the latter Applet within the execution location.

12. An end user system as claimed in any one of the preceding claims, characterised in that the end user system is a public access service provider.

13. A method for operating an end user computer system, which method comprises importing from an external source into the end user system a data stream containing or having associated with it one or more mobile program components which it is desired to execute on the computer system, which method comprises:

a. executing the mobile program component(s) within one or more program execution locations selectively isolated from or within the end user system so as to interact with data from the end user system and/or from the external source;

b. passing program components which operate in a specified manner and/or the resultant data from such interaction to the end user system via an interface which permits the transmission of data and/or program components, which correspond to specified criteria, to or from specified locations in the end user system and/or in the external source.

14. A method as claimed in claim 13 for operating an end user computer system 1, characterised in that the incoming data stream is screened to identify mobile program components present in or associated with that data stream and a selected some or all of the mobile program components are passed to one or more program execution locations selectively isolated from, or isolated within, the end user system prior to being executed to operate in a desired manner.

15. A method as claimed in either of claims 13 or 14, characterised in that the incoming data stream is written in JAVA or JAVASCRIPT language and contains or is associated with one or more Applets to be imported to the execution location, and the operating program of the execution location and/or the end user system operating program identifies the address of the imported Applet and creates a new or translated address for that Applet

corresponding to the address of a proxy Applet within the end user system, whereby when the end user system is actuated to download the imported Applet it will address and implement the proxy Applet to operate on data in the incoming data stream and/or held in the end user system and will interface with the imported Applet in the execution location.

16. A program for operating a location for the execution of at least one mobile program component in or associated with an incoming data stream imported to an end user computer system, one or more of which mobile program components are to be executed on the end user computer system, which program causes:

a. at least a selected some of the incoming mobile program components to be directed to at least one execution location which is selectively isolated from the downstream end user system; and

b. the selected mobile program component to be executed within the execution location so as to interact with data from the end user system and/or from the external source; and

c. program components which operate in a specified manner and/or the resultant data from such interaction to be passed to the end user system via an interface which permits the transmission of data and/or program components which correspond to specified criteria to or from specified locations in the end user system and/or in the external source.

17. An execution location for use in the end user system of claim 1, characterised in that it comprises a computer mechanism adapted to act as an isolating interface between an external source of data and program components and an end user computer system which is to receive or transmit data to and from the computer mechanism, which computer mechanism is programmed to divert and/or retain at least

a selected some of the program components from said external source to said computer mechanism and to execute those selected program components within the said computer mechanism and to permit transfer of data and/or program components to said end user system which operate or have been operated on in selected and specified manners.

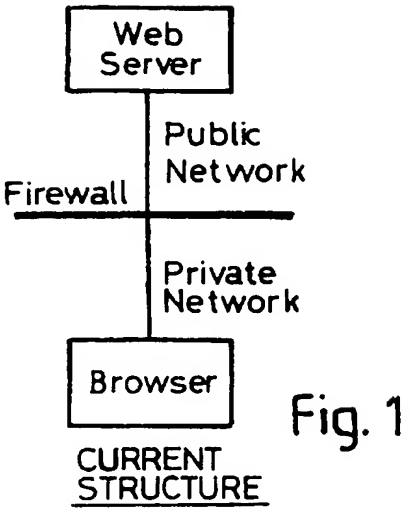


Fig. 1

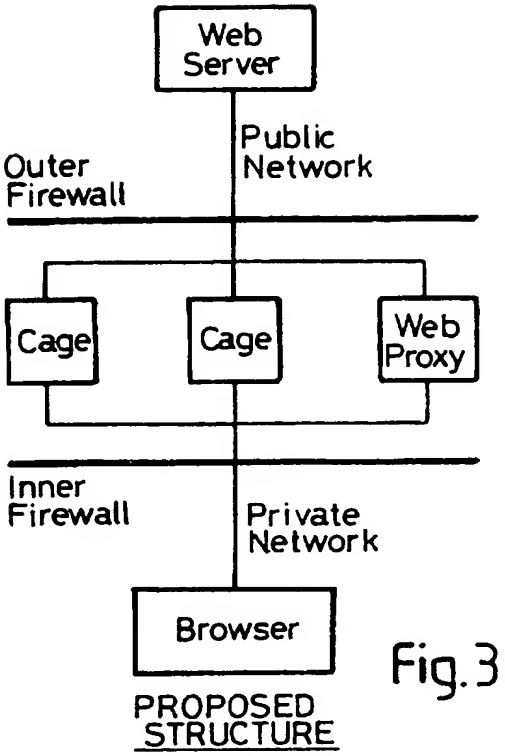


Fig. 3

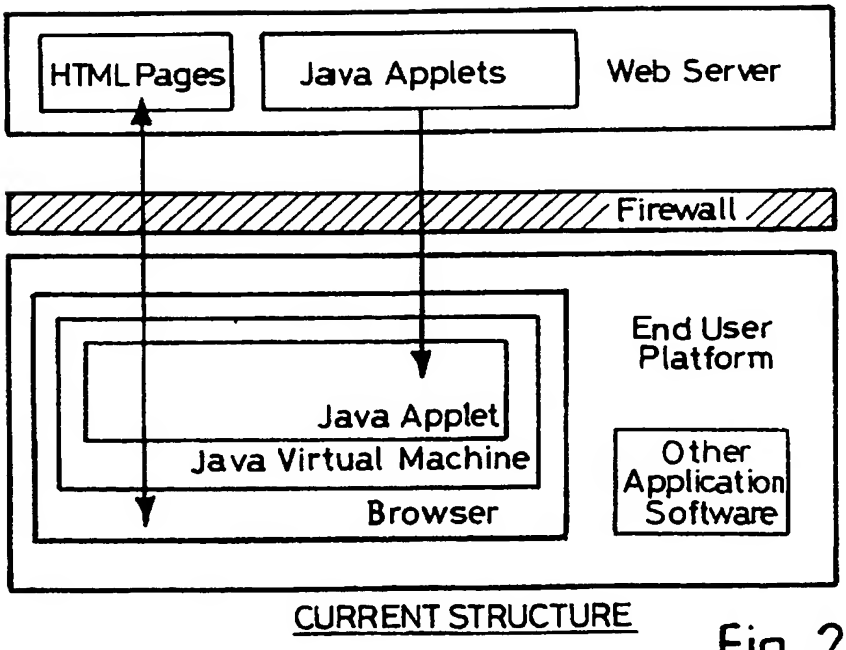
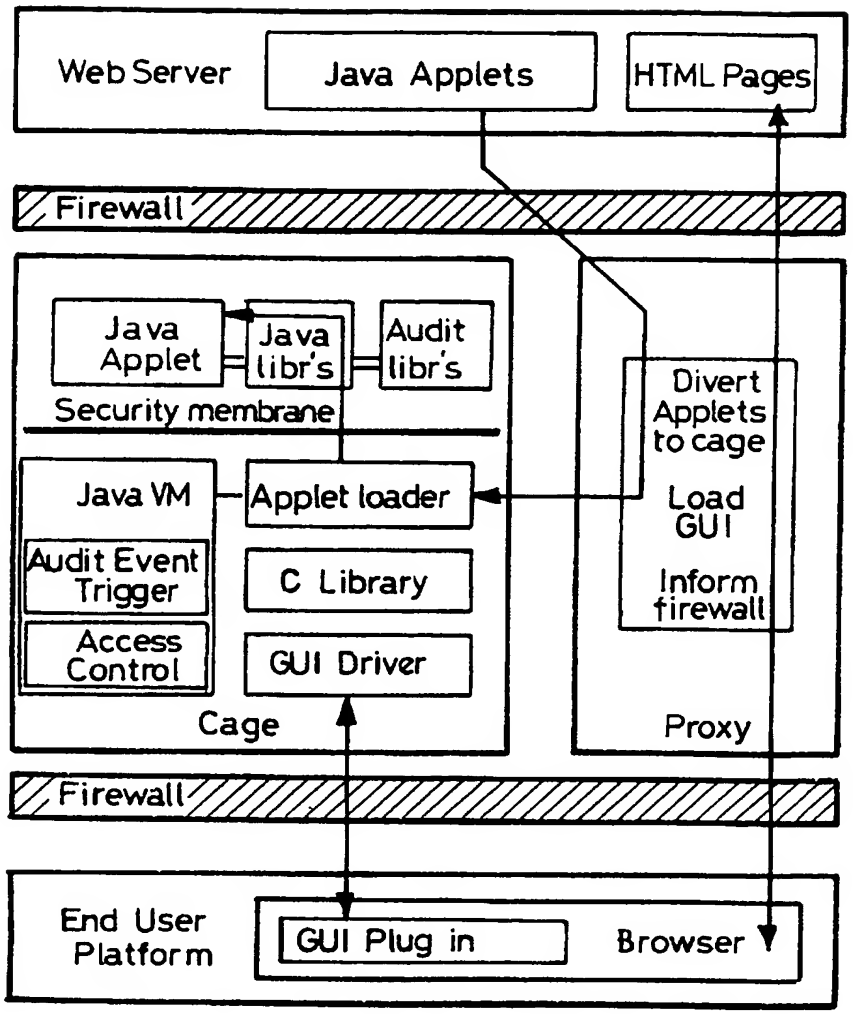


Fig. 2



Proposed Structure

Fig. 4

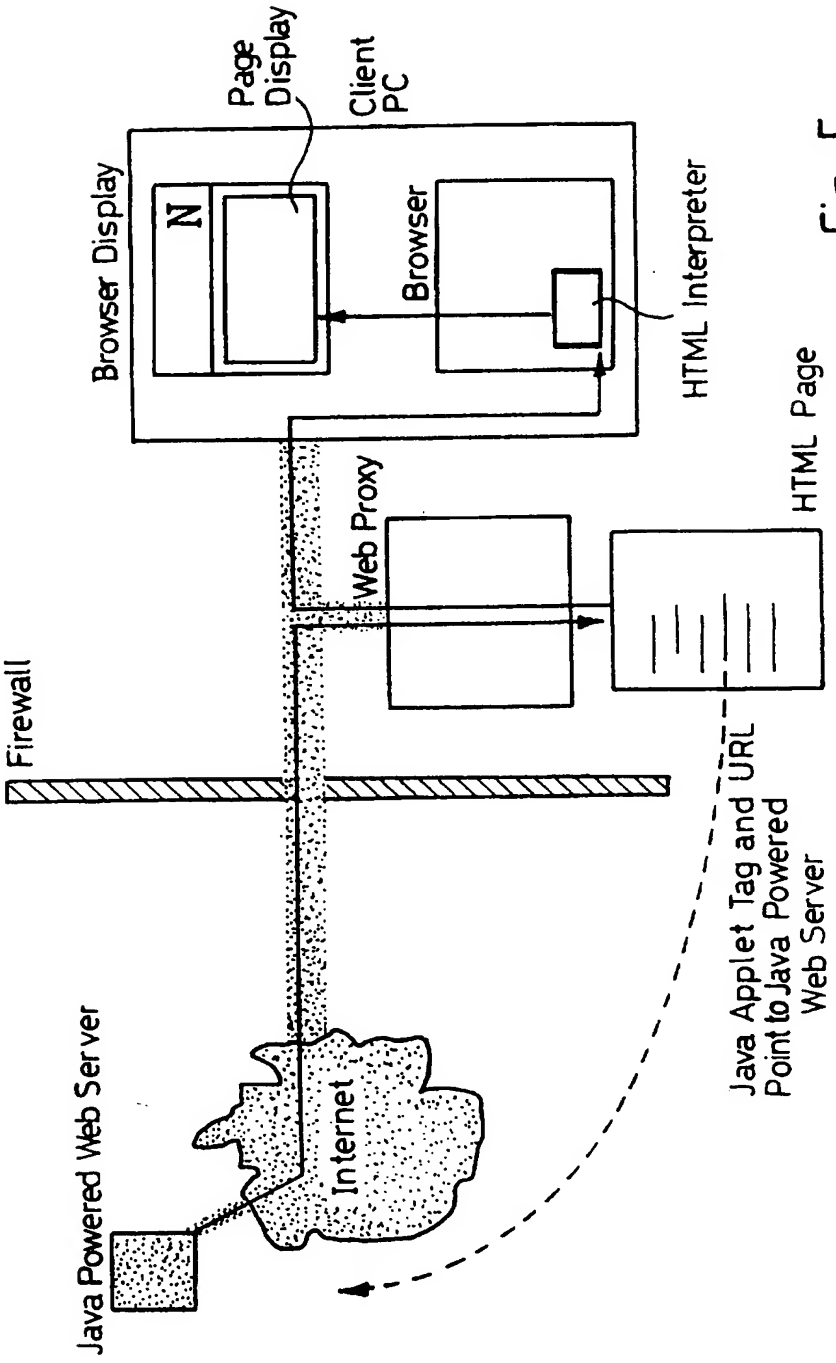

Page Display
Client PC
Browser Display
N
Browser
HTML Interpreter
HTML Page
Web Proxy
Firewall
Java Applet Tag and URL Point to Java Powered Web Server
Internet
Java Powered Web Server

Fig. 5

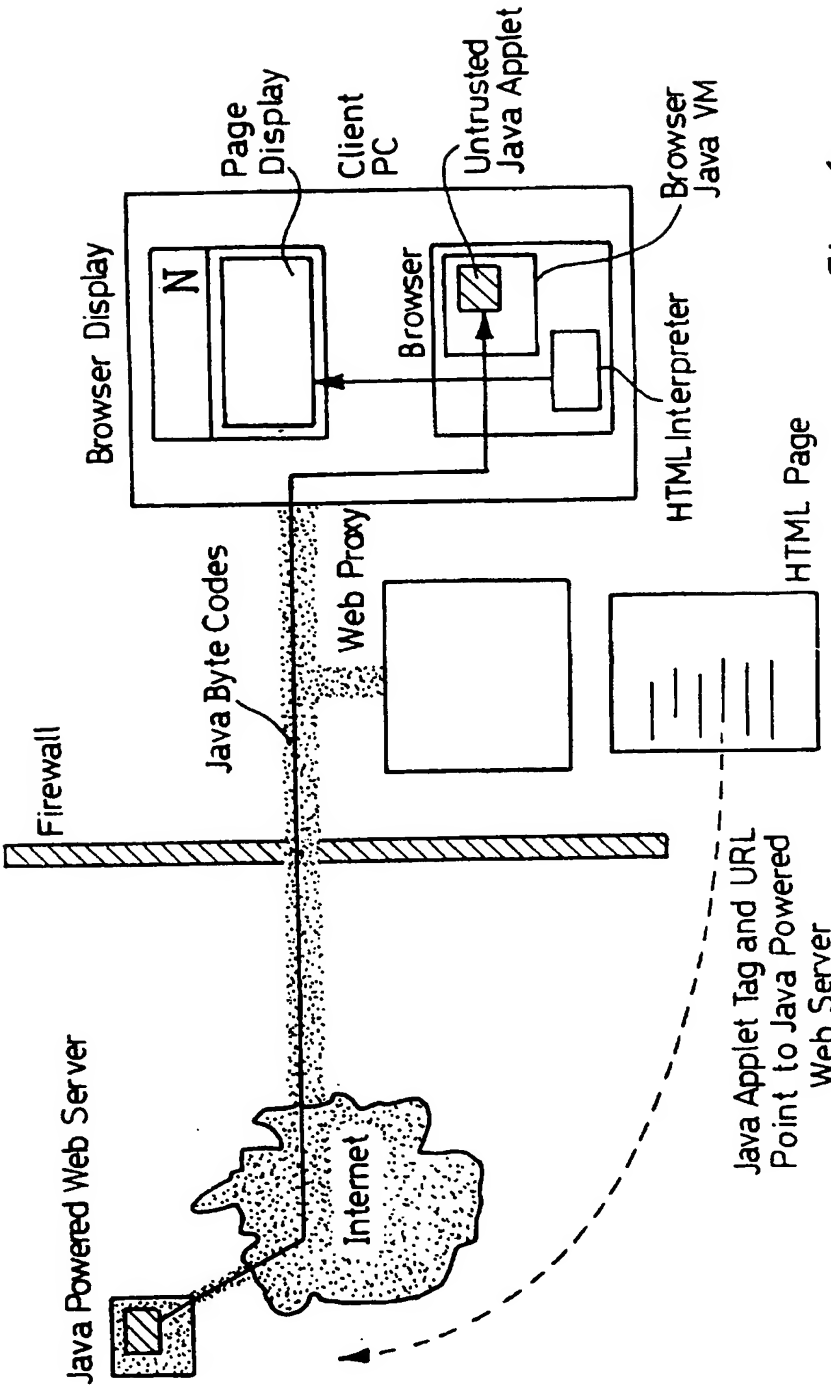

Page Display
Client PC
Untrusted Java Applet
Browser Java VM
Browser Display
Browser
HTMLInterpreter
HTML Page
Web Proxy
Java Byte Codes
Firewall
Java Applet Tag and URL Point to Java Powered Web Server
Internet
Java Powered Web Server

Fig. 6

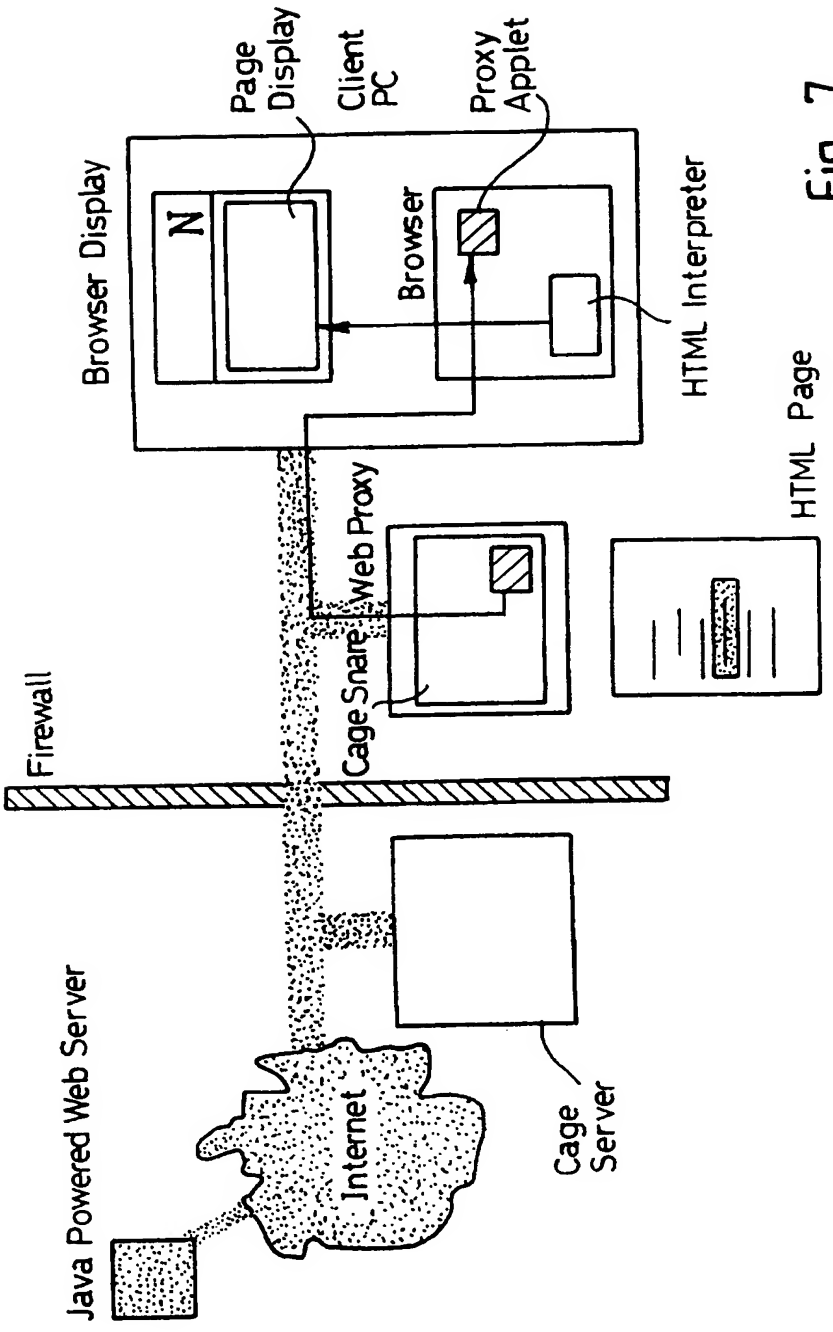

Page Display
Client PC
Proxy Applet
Browser Display
N
Browser
HTML Interpreter
HTML Page
Web Proxy
Cage Snare
Firewall
Java Powered Web Server
Internet
Cage Server

Fig. 7

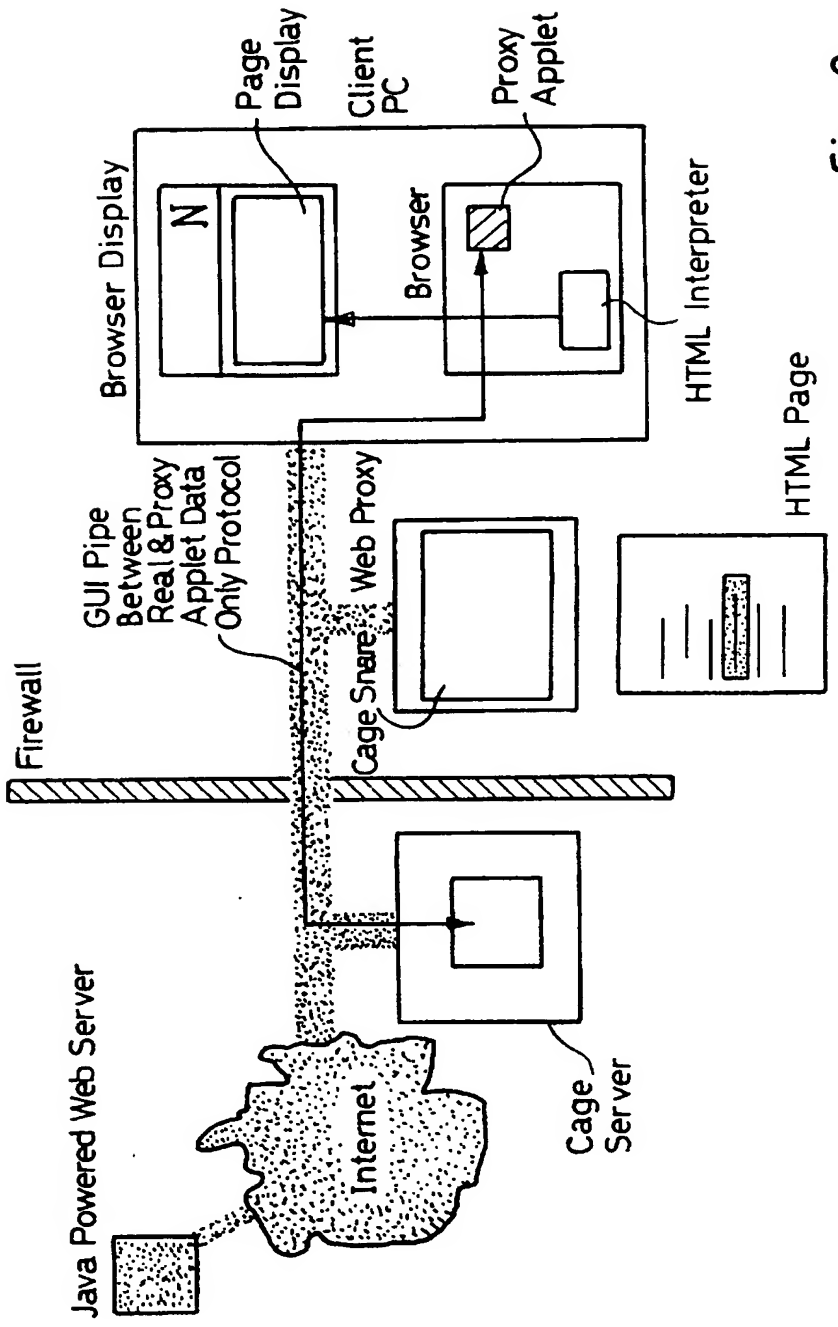

Page Display
Client PC
Proxy Applet
Browser Display
N
Browser
HTML Interpreter
HTML Page
GUI Pipe Between Real & Proxy Applet Data Only Protocol
Web Proxy
Cage Share
Firewall
Java Powered Web Server
Internet
Cage Server

Fig. 8

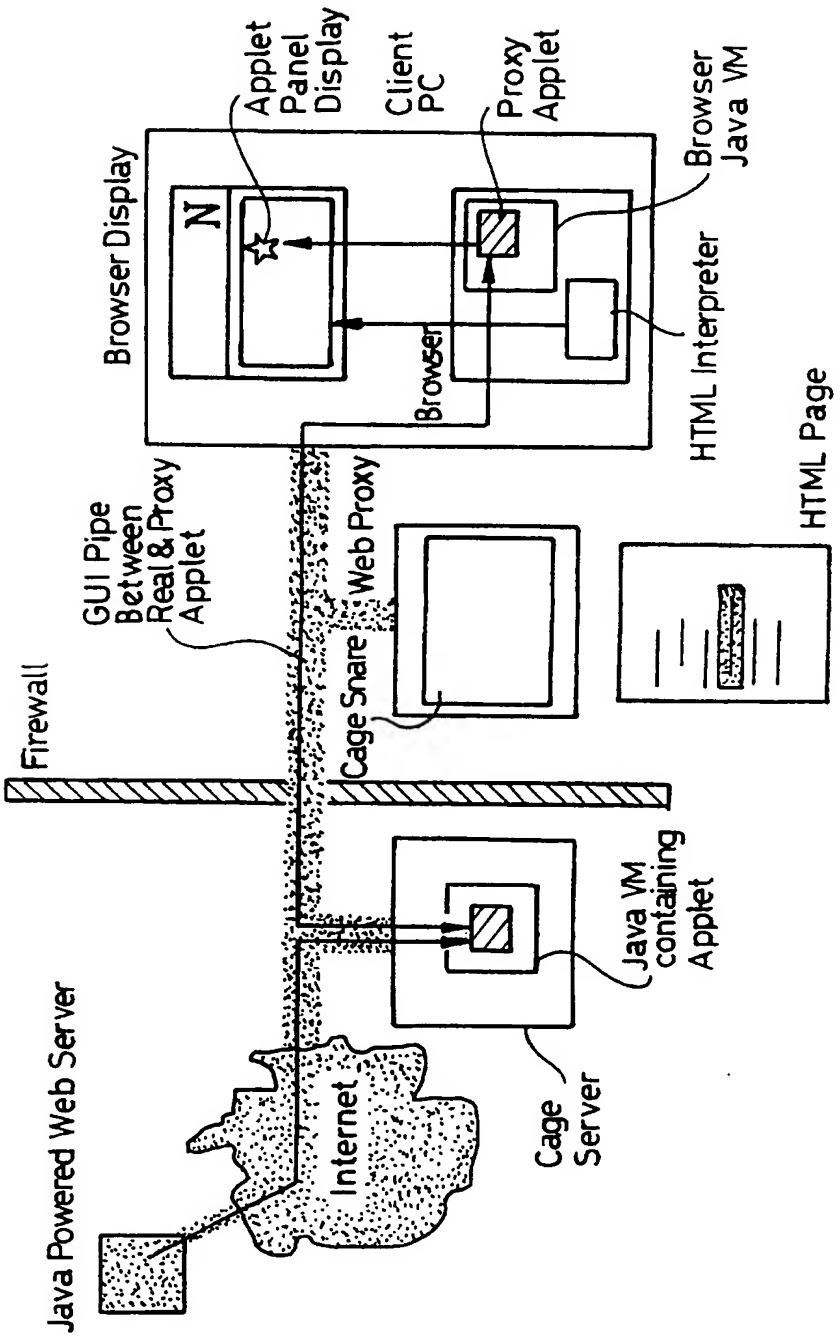

Applet Panel Display
Client PC
Proxy Applet
Browser Java VM
Browser Display
HTML Interpreter
Browser
HTML Page
GUI Pipe Between Real & Proxy Applet
Web Proxy
Cage Snare
Firewall
Java VM containing Applet
Cage Server
Internet
Java Powered Web Server

Fig. 9

INTERNATIONAL SEARCH REPORT

Interna ł Application No
PCT/IB 97/00973

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/46 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	VITEK J ET AL: "Security and communication in mobile object systems" , MOBILE OBJECT SYSTEMS. TOWARDS THE PROGRAMMABLE INTERNET. SECOND INTERNATIONAL WORKSHOP, MOS '96. SELECTED PRESENTATIONS AND INVITED PAPERS, MOBILE OBJECT SYSTEMS. TOWARD THE PROGRAMMABLE INTERNET. SECOND INTERNATIONAL WORKSHOP, MOS'96, LINZ, AUSTRIA , ISBN 3-540-62852-5, 1997, BERLIN, GERMANY, SPRINGER-VERLAG, GERMANY, PAGE(S) 177 - 199 XP002046184	1-4,7,9, 10, 12-14,16
Y	see page 181, line 1 - page 182, line 17 see page 188, line 1 - line 21 see page 190, line 13 - line 18	8
	--- -/--	

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
7 November 1997	19. 11. 97
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	**Authorized officer** Kingma, Y

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Interna il Application No
PCT/IB 97/00973

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	DEAN D ET AL: "Java security: from HotJava to Netscape and beyond" , PROCEEDINGS 1996 IEEE SYMPOSIUM ON SECURITY AND PRIVACY (CAT. NO.96CB35924), PROCEEDINGS 1996 IEEE SYMPOSIUM ON SECURITY AND PRIVACY, OAKLAND, CA, USA, 6-8 MAY 1996 , ISBN 0-8186-7417-2, 1996, LOS ALAMITOS, CA, USA, IEEE COMPUT. SOC. PRESS, USA, PAGE(S) 190 - 200 XP000634844 see abstract; figure 4 ---	8
A	EP 0 658 848 A (SUN MICROSYSTEMS INC) 21 June 1995 see the whole document ---	1,3,4,9, 13,16,17
A	ABRAMS M D ET AL: "TRUSTED SYSTEM CONCEPTS" COMPUTERS & SECURITY INTERNATIONAL JOURNAL DEVOTED TO THE STUDY OF TECHNICAL AND FINANCIAL ASPECTS OF COMPUTER SECURITY, vol. 14, no. 1, 1 January 1995, pages 45-56, XP000497494 see page 51, left-hand column, line 36 - right-hand column, line 30 ---	1,13,16
A	ROUAIX F: "A Web navigator with applets in Caml" COMPUTER NETWORKS AND ISDN SYSTEMS, vol. 28, no. 11, May 1996, page 1365-1371 XP004018234 see the whole document -----	1,13,16

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

information on patent family members

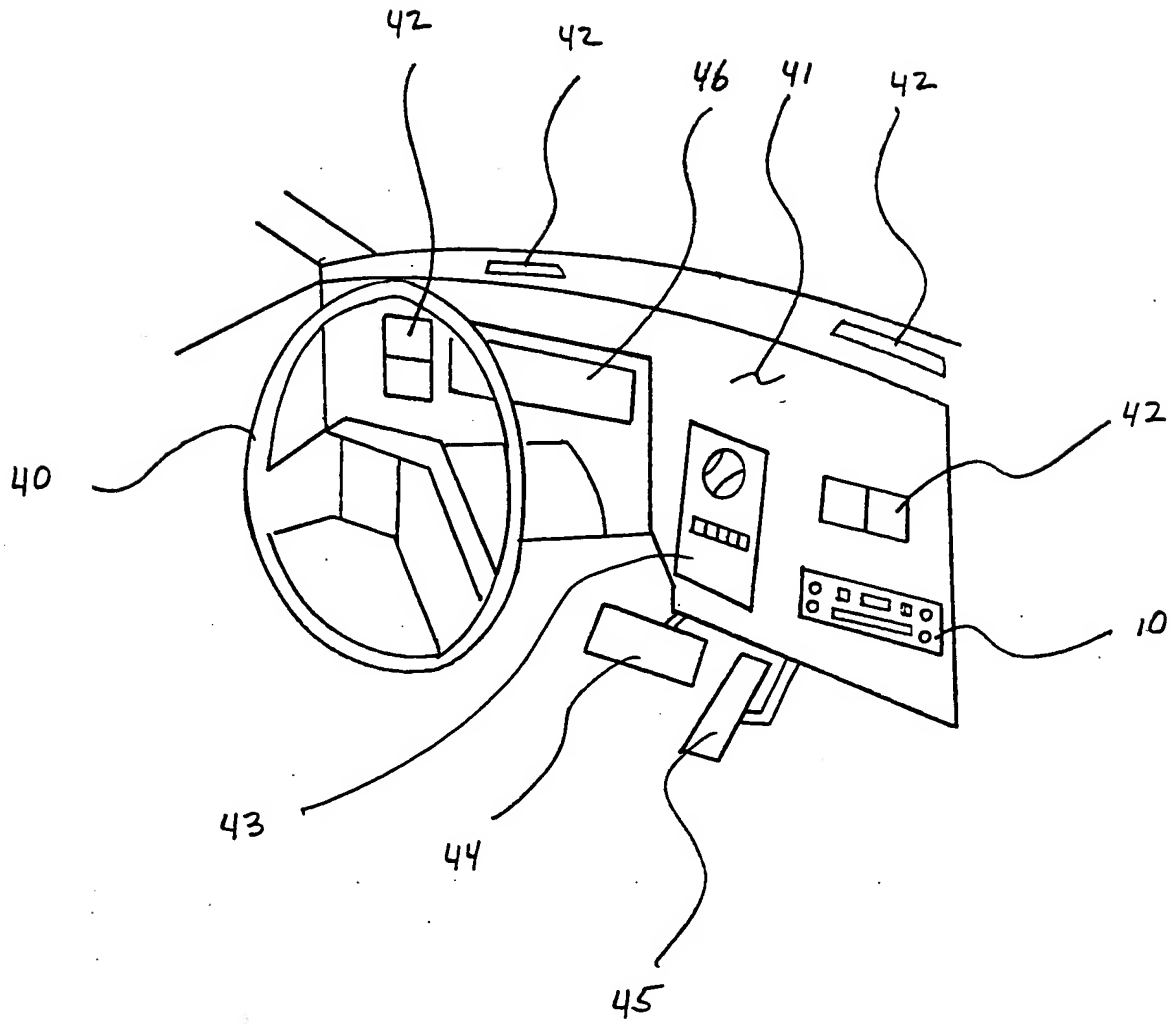
Interna al Application No
PCT/IB 97/00973

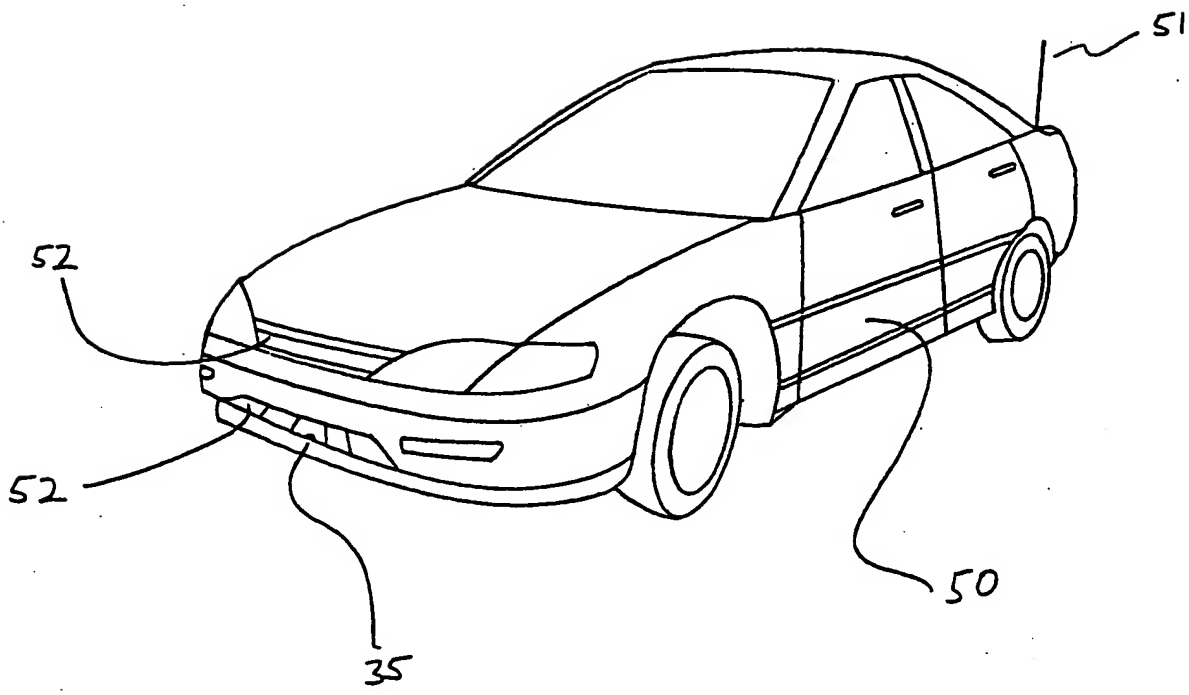
Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0658848 A	21-06-95	US 5481715 A	02-01-96
		JP 7234846 A	05-09-95

OIPE JC17
NOV 29 2004
PATENT & TRADEMARK OFFICE

10/944,405

copy




51
52
52
50
35

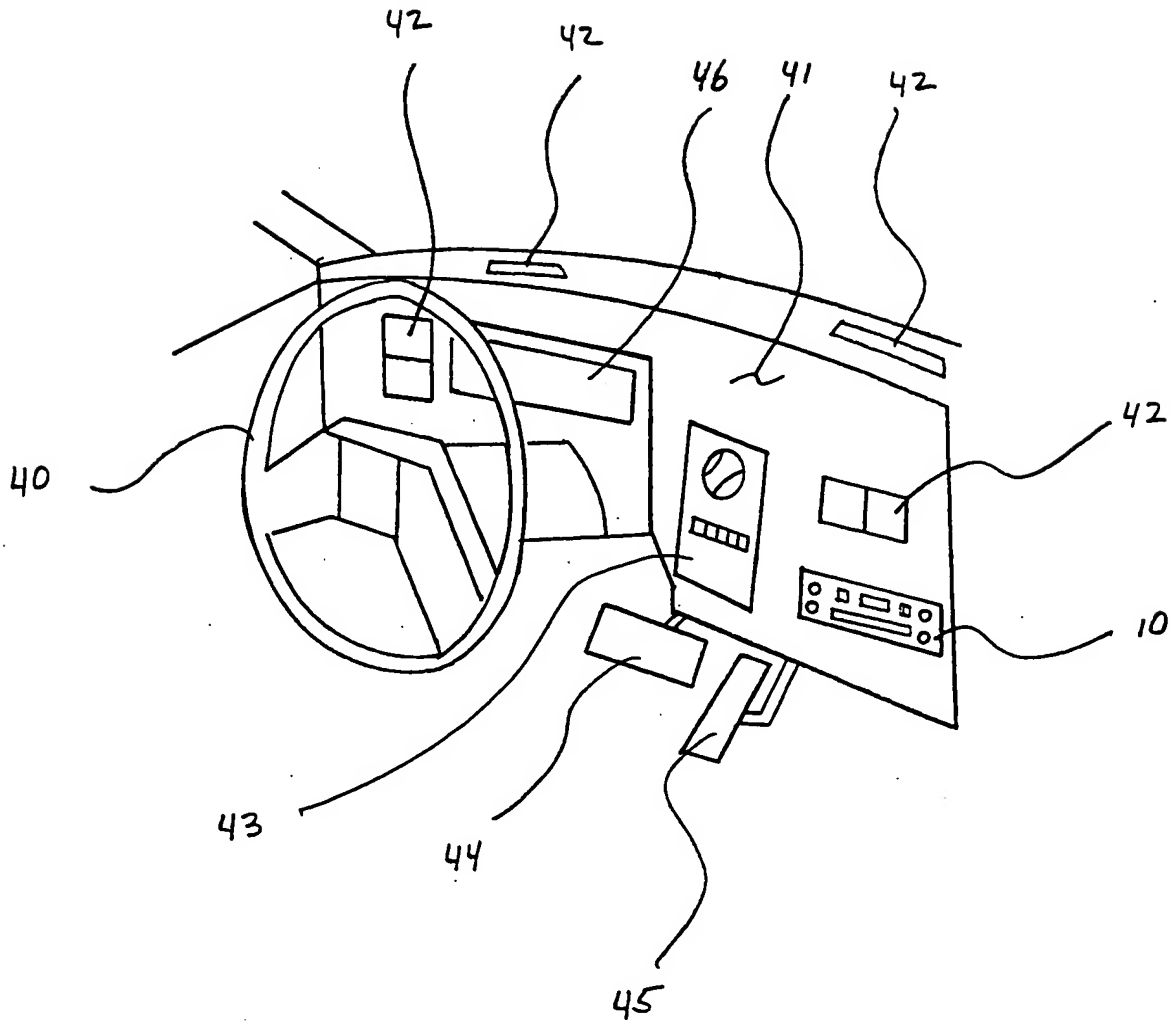


FIG 4

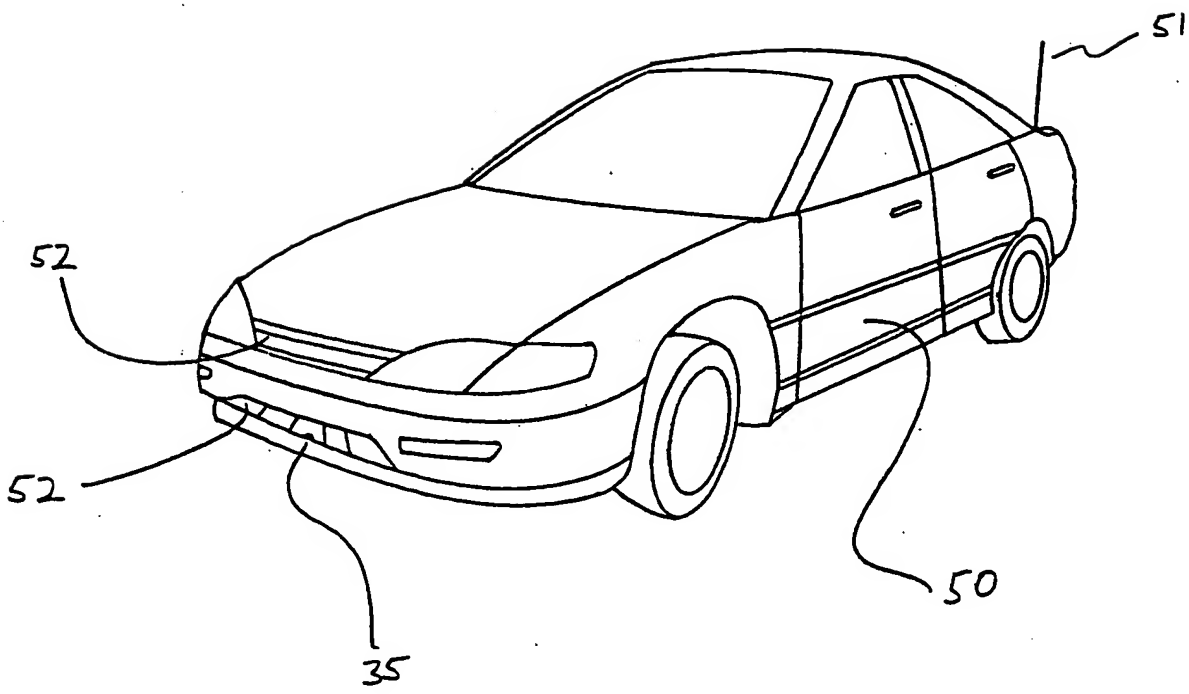


FIG 6